Dynamic Programming based Computationally Distributed Kinematic Inversion Technique

Giuseppe Casalino, Alessio Turetta, Andrea Sorbara

DIST-University of Genova
Via Opera Pia 13, genova, ITALY
{casalino,turetta,andrea.sorbara}@dist.unige.it

INTRODUCTION

The paper considers complex robotic structures characterized by the presence of a totally distributed embedded control system. More specifically, every joint is assumed to be equipped with a local “processing and communication unit” (PCU) implemented on a low-cost control device, such as a micro-controller or a FPGA. Every PCU is interfaced with both the joint sensory and actuation system, and with a communication line running inside the kinematic chain and allowing data exchanges between all the units. All the PCUs are identical and devoted, each one, to properly drive the motion of its associated joint, thus allowing to consider each element joint+link+PCU as a “1-dof-only” atomic manipulator equipped with its own embedded control system.

In this framework, the problem of controlling the operational space motion of the overall resulting kinematic structure (be it a serial chain or even a tree-structured one), can be therefore formulated in terms of decentralized control cooperation. All the atomic manipulators have to coordinate their motion in order to guarantee that the end-effectors of the overall structure reach any assigned position or track any given trajectory.

In this perspective the paper proposes an effective distributed kinematic inversion technique that, based on a finite step LQ dynamic programming technique (on-line running along the chain), automatically induces a global self-organizing control behavior among the PCUs, thus leading to the overall task execution. Motion control in the operational space is obtained by solely exploiting the control capabilities offered by the processing units located nearby every joint while not requiring any global knowledge about the overall resulting structure.

The distributed implementation of such dynamic programming algorithm is made feasible as the result of both the limited set of information received by each PCU also results into a sufficient set for allowing it computing its control action in an optimal (or quasi optimal) way, this consequently leads to a global optimal (quasi optimal) execution of the assigned overall task, which is performed without requiring any centralized a-priori knowledge of the overall structure reach any assigned position or track any given trajectory.

In addition, since the finite set of information received by each PCU also results into a sufficient set for allowing it computing its control action in an optimal (or quasi optimal) way, this consequently leads to a global optimal (quasi optimal) execution of the assigned overall task, which is performed without requiring any centralized a-priori knowledge of both the geometry and kinematics of the overall structure.

The paper is here intended as a new contribution to the whole field of modular robotic systems, in the sense that it allows to enlarge the original modularity concepts (mechanical, electrical and joint actuation) intensively dealt within the literature, till the operational space control level; being this last an issue that, at the best of authors knowledge, seems having been very little considered till recent years and however, when seldom considered, always referred to centralized approaches to the control and coordination of the various composing modules.

Earlier modular robotic structures are indeed typically governed by ad-hoc centralized controllers, specifically suited to the characteristics and physical parameters of the considered system. By adopting the proposed decentralized approach to coordination, every basic robotic unit is instead able to operate in autonomy even when unplugged from the other ones, as an independent stand-alone robotic system. Once it is plugged together with other units, a common communication channel can be automatically established, linking all the PCUs and allowing them to exchange information necessary for cooperation. In this way the coordination functionality, rather than being totally provided by an additional centralized processing unit, a-priori aware of the kinematic and dynamic parameters of the overall system, can be obtained, in a distributed fashion, as a result of the sole data on-line exchanged among all the PCUs. Every PCU has just to know the structural parameters of its associated basic robotic unit and can be totally unaware of the characteristics of the overall robotic structure. All the data needed for enabling its corresponding basic robotic unit to act cooperatively with the other ones are on-line obtained through the communication channel. As a matter of fact, by adopting such a strategy it will be possible to implement a real plug’n’play behavior, since every robotic module, will be immediately able to interact and cooperate with any other one, just upon their electro-mechanical interconnection.

The paper will be organized as follows. The starting point will be the description of the details of the proposed technique when adopted to control and coordinate open serial kinematic chains. The obtained results will be then extent to deal with open tree-structured chains. Finally the DP-based technique will be also applied to the more complex case of tree-structured chains constrained by the presence of a grasped rigid object.
LINEAR KINEMATIC CHAINS

Consider a generic open linear kinematic chain, as the one in Fig. 1-a, where \( <o> \), \( <e> \), \( <t> \) respectively denote the absolute-frame, the end-effector frame and a generic tool frame rigidly attached to \( <e> \). Also assume that at the beginning of every sampling interval a forward-backward pipelined exchange of geometric data is first performed along all the PCUs of the kinematic chain. More details on such a preliminary phase are reported in [13] and therefore will not be here repeated. For the goal of the current work it is sufficient to say that, at the end of this very quick data exchange phase, every PCU knows both the current position of its corresponding joint+link w.r.t. \( <o> \) frame, and the actual position of \( <e> \) w.r.t. \( <o> \). By exploiting such information, every \( i \)-th PCU is hence in the condition to evaluate the Jacobian 1-column matrix \( h_i \), describing the contribution to the motion of \( <e> \) frame provided by the associated \( i \)-th joint, and actually corresponding to the \( i \)-th column of the Jacobian matrix \( J \) of the overall structure

\[
J = [h_1, h_2, ..., h_n]
\]

On the basis of such a limited local information set (the sole \( h_i \) vector), every PCU can then, in the second part of every sampling interval, cooperate with the other ones in order to solve, in a distributed way, the overall chain kinematic inversion problem, by employing the announced DP based technique that will be hereafter presented.

To this aim, for sake of clearness, the simpler case of a 6-component task defined in terms of both angular and linear desired velocities for the end-effector frame will be first considered, as a representative one; the obtained results will be later extended, in the second part of the section, to cope also with more general tasks, possibly referred to other frames (for example \( <t> \)) and/or differing in dimension (not necessarily equal to 6). Finally, some remarks regarding both the obtained solutions will be drawn in the third part of the section.

a) Execution of a 6-component end-effector velocity task

Consider first the problem of making the end-effector generalized velocities vector \( \dot{x} \equiv [\omega^T, v^T]^T \) maximally close to a desired value \( \ddot{x} \) and observe how such a problem can actually be formulated as that of finding (generally one of) the solutions of the following quadratic optimization problem:

\[
\min_{\dot{q}} \| \ddot{x} - \dot{x} \|^2 \quad (2)
\]

where \( \dot{q} \) is the joint velocity vector producing the actual end-effector velocity \( \dot{x} \), via the well known Jacobian relationship

\[
\dot{x} = J \dot{q} \quad (3)
\]

Then note how the direct kinematic relationship (3) can also be interpreted in terms of the following \( n \)-step linear dynamic system (obviously assumed completing its \( n \) steps run within a single sampling interval)

\[
\dot{x}_i = \dot{x}_{i+1} + h_i \dot{q}_i \ ; \ \dot{x}_n = 0 \ ; \ i = 1,2,..,n \ ; \ \dot{x} = \dot{x}_n \quad (4)
\]

Thus showing that optimization problem (2) can also be interpreted as a classical “final-state, cheap-control” LQ optimization problem established along (4); therefore solvable via the application of very well known dynamic programming techniques (see among the others [14]).

To this aim, first represent (4) with respect to \( \ddot{x} \), by introducing the “error” vectors

\[
\dot{\varepsilon}_i = (\ddot{x} - \dot{x}_i) \ ; \ i = 1,2,..,n
\]

Fig. 1 – a) Linear kinematic chain. b) Tree structured kinematic chain. c) Constrained kinematic chain
thus consequently obtaining
\[ \dot{\hat{q}}_i = \dot{\hat{q}}_{i-1} - h_i \hat{q}_i \quad ; \quad \dot{\hat{q}}_n = \ddot{x} \quad ; \quad i = 1, \ldots, n \] (6)

Then, decompose problem (2) (once rewritten accordingly with position (5)) into the following sequence of optimal cost-to-go problems
\[ \min_{\tilde{q}} \left\| \sum_{i=1}^{n} \left( \tilde{q}_i \right)^2 \right\| = \left\| \tilde{q}_{n-1} \right\|^2 \quad ; \quad i = n, \ldots, 1 \] (7)

It then follows that, by backward solving (7) as here indicated
\[ \left\| \tilde{q}_{n-i} \right\| = \min_{\tilde{q}_i} \left\| \tilde{q}_{n-1} \right\|^2 \quad ; \quad i = n, \ldots, 1 \] (8)

the solution of the associated overall optimization problem can be obtained via the following regularized version of the well known DP dual-step procedure

**Bakward phase** \( (i = n, \ldots, 1) \)
\[ V_{n+1} = I \quad ; \quad p_i = f(h^T_i V_{i+1} V_{i+1}^T h_i) \geq 0 \quad ; \quad l_i^T = (p_i + h^T_i V_{i+1} V_{i+1}^T h_i)^{-1} h^T_i V_{i+1} V_{i+1} ; \quad V_i = V_{i+1} (I - h_i l_i^T) \] (9)

where \( f(.) \) (acting as regularizing parameter) is a bell shaped, finite support, scalar positive function, attaining its maximum value in the origin of its argument.

**Forward phase** \( (i = 1, \ldots, n) \)
\[ \dot{\hat{q}}_n = \ddot{x} \quad ; \quad \dot{\hat{q}}_1 = V_{i+1}^T h_i \hat{q}_i \quad ; \quad \dot{\hat{q}}_i = \dot{\hat{q}}_{i-1} - h_i \dot{\hat{q}}_{i-1} \] (10)

Before closing the subsection it is worth noting how forms (9) and (10) differ from the standard solution of a generic “final state, cheap control” LQ optimization problem for just the presence, at every \( i \)-th stage, of the regularizing parameter \( p_j \). As better described in the third sub-section, such a term may affect the optimality of the resulting control law (10); nevertheless its introduction has been actually necessary for preventing that the corresponding joint velocity reference \( \dot{\hat{q}}_i \) grow toward unacceptable high values, whenever the term \( (h^T_i V_{i+1} V_{i+1}^T h_i) \) occasionally attains nearly-zero values.

**b) Execution of a generic velocity task**

Consider now, for sake of generality, the following linear transformation of the end-effector velocity vector
\[ \dot{\Theta} = H \dot{x} \quad ; \quad H \in \mathbb{R}^{m \times 6} \] (11)

where \( m \) (i.e. the task dimension) may alternatively be **equal** to 6 (as in the case of a rigid body velocity transformation such as, for example, the one existing from \( <e> \) to \( <t> \)), **lower** than 6 (for instance to refer to the sole angular velocity vector \( \omega \), or to the linear velocity \( v \) only), or even **greater** than 6 (as in the case described in the next section).

Then, analogously to what seen before, consider the problem of finding the joint velocity references which make \( \dot{\Theta} \) maximally close to a desired value \( \ddot{\Theta} \)

\[ \min_{\tilde{\dot{\Theta}}} \left\| \tilde{\dot{\Theta}} - \dot{\Theta} \right\|^2 = \min_{\tilde{\dot{\Theta}}} \left\| \tilde{\dot{\Theta}} - H \ddot{x} \right\|^2 \] (12)

To approach such a problem, first represent reference vector \( \tilde{\dot{\Theta}} \) via the projected form
\[ \tilde{\dot{\Theta}} = H \ddot{x} + d \] (13)

with
\[ \ddot{x} = H^\# \tilde{\dot{\Theta}} + (I - H^\# H) \ddot{z} \quad ; \quad d = (I - H H^\#) \tilde{\dot{\Theta}} \perp \text{Span}(H) \] (14)

and where \( \ddot{z} \) is any finite arbitrary vector. Then, by exploiting expression (13), the following equivalent representation of (12) can be obtained

\[ \min_{\tilde{\dot{\Theta}}} \left\| H(\ddot{x} - \dot{z}) \right\|^2 + \left\| d \right\|^2 \] (15)

where the decomposition into the summation of two separate squares is made possible by the orthogonal property expressed by the second of (14); while the extraction of the second term from the minimum operation is instead allowed by its independence from \( \ddot{x} \) and then from \( \dot{z} \). As a consequence, the solutions of the original optimization problem (12) certainly must coincide with those of the following one.
\[
\min_q \left\| H(\tilde{x} - \hat{x}) \right\|^2 = \min_q \left\| R(\tilde{x} - \hat{x}) \right\|^2
\]

(16)

where the 6x6 matrix \( R \), resulting from the following SVD based factorization process¹

\[
Q \triangleq H^TH \geq 0 \Rightarrow Q = R^TR \geq 0
\]

(17)

has been introduced, in lieu of \( H \), to always guarantee the presence of a squared matrix inside the cost expression (notwithstanding with the value of \( m \)).

Since expression (16) has a similar form w.r.t. problem (2), it can be solved by employing the same dual step procedure seen before, thus obtaining again a linear control law, with the only difference that now

\[
l_i^T = (p_i + h_i^T T_{r_i} T_{r_{i+1}} \dot{h}_i)^T h_i^T T_{r_i} T_{r_{i+1}} ; \quad i = n, ..., 1
\]

(18)

where

\[
T_i = RV_i ; \quad i = n + 1, ..., 1
\]

(19)

c) Remarks

Since problem (2) is actually a special case of more general problem (16), the following considerations, will refer to the latter one only, while however holding also for the former one.

1. Despite terms \( p_i \) are certainly necessary for regularization purposes, a resulting drawback is however implied by their presence, due to the fact that, whenever at least one of them is not null, a sort of “perturbation” is introduced in the (otherwise optimal) control action (18). As a consequence, cost (12) may be generally prevented from the exact achievement of its minimum attainable value. Nevertheless, it can be shown that by letting

\[
r \triangleq \max_q \delta(HJ) = \max_q \delta(RJ) \leq 6
\]

(20)

be the maximum rank of the “task-associated” Jacobian matrix, no optimality character is lost, provided \( r \) joints are found along the chain, exhibiting each one a null regularizing \( p_i \) term. In other words and consistently with intuition, if a number of \( r \) control units works in an optimal way, the maximum possible fulfillment of the assigned task is guaranteed and therefore the proposed solution is still an optimal one. In addition, if in such circumstances the task dimension \( m \) is not greater than \( r \), (thus implying that the assigned task can be totally accomplished by the system), the resulting cost is zeroed, thus leading to the complete achievement of the reference velocity vector.

2. By backward exploring the last expression of (9) it is easy to see that the following fact holds

\[
V_i = \prod_{i=n}^1 (I - h_i^T) \quad \Rightarrow \quad V_i (\dot{\hat{x}} - \hat{x}_o) = \dot{\hat{x}} - \hat{x}
\]

(21)

As a consequence, through the knowledge of \( V_i \) matrix and the base velocity vector \( \hat{x}_o \) (in case not null), it is possible, since the end of the backward stage, to determine which will be the actual end-effector velocity resulting from any given reference \( \dot{\hat{x}} \). A property this last which will reveal to be of fundamental importance in section 4 for dealing with constrained chains.

3. Finally, as it concerns implementation aspects, it should be evident how the proposed backward-forward procedure is natively suitable for distributed pipelined computations. To this respect, it is also worth noting how simple result to be both the computation and communication burdens required to every PCU (see Fig. 2). Further implementation details can however be found in [13].

Fig. 2 – Single PCU distributed kinematic inversion scheme

¹This can be done, for instance, via the use of SVD algorithms as follows: \( A \triangleq V^T \Sigma V = (V^T \sqrt{\Sigma}) (\sqrt{\Sigma} V)^T = R^TR \)
TREE-STRUCTURED KINEMATIC CHAINS

Let us now consider a tree-structured kinematic chain, as for instance the one reported in fig. 1-b, where each branch is now regarded as a linear sub-chain separately controlled by the same DP procedure described in the previous section. By now assuming that desired velocity tasks \( \dot{\theta}_{1}, \dot{\theta}_{2} \) have been assigned to the two upper-lying sub-chains, the overall cost to be minimized consequently turns out to be represented by the following summation

\[
\min_{\dot{q}_{1}, \dot{q}_{2}, \dot{q}_{3}} \left( \sum_{i=1}^{2} \left\| \dot{\theta}_{i}^{-1} \dot{\theta} \right\|^{2} \right) = \min_{\dot{q}_{1}, \dot{q}_{2}, \dot{q}_{3}} \left( \sum_{i=1}^{2} \left\| \dot{\theta}_{i}^{-1} \dot{\theta} \right\| \right) = \min_{\dot{q}_{1}, \dot{q}_{2}, \dot{q}_{3}} \left( \sum_{i=1}^{2} \left\| \dot{\theta}_{i}^{-1} \dot{\theta} \right\| \right)
\]

where the optimality principle has been formerly applied at the “macro-level” of the composing sub-chains; and the resulting cost-to-go successively split into two separate ones, due to the resulting independence (for any given \( \xi \)) of the upper-lying sub-chains.

To solve problem (22), first “internally” apply the dynamic programming technique to every unit of the upper-lying sub-chains till arriving to the so called “Y-node” junction, which represents the mechanical connection between the three sub-chains. So doing, while keeping into account that now the basis of the upper-lying sub-chains are both subjected to the common velocity \( \xi \), provided by the end-effector \( e_{3} \) of the under-lying chain) we get the form

\[
\min_{\dot{q}_{1}, \dot{q}_{2}, \dot{q}_{3}} \left( \sum_{i=1}^{2} \left\| \dot{T}_{i} (\dot{\xi} - S^{\dot{\xi}} \dot{x}) \right\| \right)
\]

where 6x6 non-singular matrices \( T_{i}; i = 1, 2 \) are the rigid-body velocity transformations currently established from the end-effector frame \( e_{3} \) of the underlying sub-chain, till the end effector-frames \( e_{i}; i = 1, 2 \), of the upper-lying ones.

The above minimization problem however needs some pre-processing, in order to be accommodated for the same dynamic programming procedure to be now internally applied along the under-lying sub-chain. To this aim, an additional PCU, located in correspondence of the “Y node”, has to be introduced, in order to perform such needed pre-processing, accordingly with the following step-by-step reasoning line:

1. Represent the cost in (23) in the equivalent form

\[
\sum_{i=1}^{2} \left\| \dot{T}_{i} (\xi - S^{\dot{\xi}} \dot{x}) \right\|^{2} = \left\| T \dot{x} - TS^{\dot{\xi}} \right\|^{2}
\]

where

\[
T = \text{diag} (\dot{T}_{1}, \dot{T}_{2}) \quad ; \quad S = \begin{bmatrix} S^{T}, S^{T} \end{bmatrix}^{T} \quad ; \quad \dot{x} = \begin{bmatrix} \dot{x}^{T}, \dot{x}^{T} \end{bmatrix}^{T}
\]

2. Further express \( T \dot{x} \) via the following projected form

\[
T \dot{x} \approx TS^{\dot{\xi}} \tilde{x} + \tilde{d}
\]

where, obviously enough

\[
\tilde{x} \doteq (TS)^{\theta} T \dot{x} + (I - (TS)^{\theta} (TS)) \tilde{d} \quad \tilde{d} \doteq (I - (TS)(TS)^{\theta}) T \dot{x} \perp \text{Span} (TS)
\]

with \( \tilde{d} \) any finite arbitrary vector.

3. Then, by substituting (26) into (24), while keeping into account the orthogonal condition in (27), consequently obtain

\[
\left\| T \dot{x} - TS^{\dot{\xi}} \tilde{x} \right\|^{2} = \left\| \dot{R} (\dot{x} - S^{\dot{\xi}} \tilde{x}) \right\|^{2} + \left\| \tilde{d} \right\|^{2}
\]

where, similarly to section 1, 6x6 matrix \( \dot{R} \) is defined as the result of the following SVD-based factorization process

\[
\dot{Q} \doteq (TS)^{\theta} (TS) \geq 0 \quad \Rightarrow \quad \dot{Q} = \dot{R}^{T} \dot{R}
\]

4. As it can be now easily noted, form (28) is just suitable to be properly managed by the same dynamic programming procedure of section 1, now applied to the under-lying sub-chain. More specifically, by keeping into account that \( \tilde{d} \) is it also independent from \( \dot{x} \), everything for the under-lying chain obviously proceeds as in section 1.

As it concerns implementation details, it should be noted how the pre-processing activities, substantially reducing to the very feasible evaluations of the sole (26) and factorization (28), necessarily require that the Y-node PCU acquire, from the lowest PCU of every upper-lying chain, the corresponding evaluated matrices \( \dot{T}_{i} \) and \( \dot{S} \), other than velocity
reference $\dot{\hat{\gamma}}$. Also note how such computations are however always of the same structure and dimensionality; thus resulting they also independent from the complexity of the connecting chains.

**CONSTRAINED TREE-STRUCTURED KINEMATIC CHAINS**

Let us now consider the case of a tree-structured kinematic chain, now constrained to manipulate a rigid object, as schematically indicated in Fig. 1-c. For sake of simplicity, the considerations that will be hereafter reported will only refer to the case of a rigid grasping performed by both the end-effectors of the upper-lying sub-chains. The more complex case of non-rigid grasping (as it is for instance the case of finger-tip manipulation performed by a generic hand-arm system) even if not deserving particular difficulties in extending the results hereafter reported, is however deferred to a forthcoming work.

With this in mind, while letting $\dot{y}$ be the body-frame <b> generalized velocity vector (projected on <o>) also observe how, in force of the established rigid-motion constraint, such body velocity results related with the end-effectors velocities via any one of the following relationships

$$\dot{y} = L_1^{-1}\dot{x} = L_2^{-1}\dot{x} \quad ; \quad L_1, L_2 \in \mathbb{R}^{6 \times 6}$$

(30)

With non-singular matrices $L_1, L_2$ representing the rigid body velocity transformations (both projected on <o>) from end-effector frames $<e_1>, <e_2>$ toward $<b>$, respectively; and where the second equality in (34) just corresponds to the mentioned rigid-motion constraint fulfilled by both the end effector velocities.

By letting vector $\dot{\theta}$ be a generic linear transformation of $\dot{y}$

$$\dot{\theta} = H \dot{y} \quad ; \quad H \in \mathbb{R}^{m \times 6}$$

and $\vec{\theta}$ be the corresponding desired reference vector, the problem of making $\dot{\theta}$ maximally satisfying the requirement can be again formulated in terms of the following LQ optimization problem

$$\min_{\dot{q},\dot{q}, \dot{q}} \left\| \vec{\theta} - \dot{\theta} \right\|^2 = \min_{\dot{q},\dot{q}} \left( \min_{\dot{q},\dot{q}} \left\| \vec{\theta} - H\dot{y} \right\|^2 \right)^2$$

(32)

now however constrained by

$$\left\{ (\dot{i}_1, \dot{i}_2) \in \Gamma \right\} = \left\{ (\dot{i}_1, \dot{i}_2) \quad \text{s.t. } (L_1^{-1}\dot{x} = L_2^{-1}\dot{x}) \right\}$$

(33)

since joint velocity vectors $\dot{i}_1, \dot{i}_2$ have to lie within the set of those guaranteeing the fulfilment of the rigid body constraint (33); vector $\dot{i}_1$ is instead unrestricted, since always inducing rigid motion contributions to the overall upper-lying structure.

As it can be easily realized, optimization problem (32) falls into the well known class of the so called “two-point-boundary-values” optimal LQ control problems, since it is subjected to constraints regarding both its overall initial state (i.e. $\dot{x}_0 = \dot{x}_0 = \dot{x}_0 = \dot{x}_0$) and its final one (i.e. constraint (33)).

Despite this fact (i.e. despite the possibility of finding an optimal solution within such class of problems), for the time being, we shall limit ourselves in proposing a slightly sub-optimal approach that, seems however exhibiting the non-negligible advantage of preserving most of the structure already devised for the unconstrained case.

To this aim, first represent $\vec{\theta}$ via its projected form

$$\vec{\theta} = H \ddot{\gamma} + \dot{d}$$

(34)

where again

$$\ddot{\gamma} = H^+ \ddot{\gamma} + (1 - H^+ H) \ddot{z} \quad ; \quad \dot{d} = (1 - H H^+ ) \ddot{\theta} \perp \text{Span}(H)$$

(35)

Then, by posing (accordingly with constraint)

$$(\dot{i}_1 \dot{i}_2 ) \ddot{\gamma} \quad ; \quad i = 1, 2$$

(36)

obtain, analogously to what done in previous sections, the following cost to be minimized, under fulfillment of (33).

$$\min_{\dot{i}_1, \dot{i}_2} \left( \min_{\dot{i}_1, \dot{i}_2} \left( \sum_{i=1}^{2} \left\| R \left( (\dot{i}_1 \ddot{x} - \dot{x}) \right) \right\|^2 \right)^2 \right)^2$$

(37)

with again

$$(HL_i)^T HL_i \geq 0 \Rightarrow \dot{i}Q = \dot{i}R^T R \quad i = 1, 2$$

(38)
At this point, by letting each upper-lying sub-chain separately apply the backward phase of its corresponding DP procedure, till arriving to the Y-node junction, the resulting cost has the same form than in the unconstrained case (23). Differing from before, however, in this case, in order to guarantee the fulfillment of the rigid motion constraint, the Y-node PCU has now to perform some different pre-processing activities, motivated by the following reasoning line.

1. First of all note that, since the pair $i \mathbf{T}^i \mathbf{T}$ per-se already satisfies a rigid motion constraint, the term $i \mathbf{T}^i \mathbf{T}$ within cost expression (23) can now more simply be decomposed as follows:

$$i \mathbf{T}^i \mathbf{T} \equiv i \mathbf{T} \mathbf{S}^3 \mathbf{x} \tag{39}$$

where we have obviously posed

$$\mathbf{S} \mathbf{x} \equiv i \mathbf{T}^{-1} i \mathbf{T} ; \quad i = 1, 2 \tag{40}$$

As a consequence, the cost to be minimized by the lower-lying sub-chain becomes:

$$\sum_{i=l} \left\| i \mathbf{T} (\mathbf{x} - \mathbf{S}^3 \mathbf{x}) \right\|^2 = \sum_{i=l} \left\| i \mathbf{T} \mathbf{S}^3 (\mathbf{x} - \mathbf{S}^3 \mathbf{x}) \right\|^2 \tag{41}$$

2. Then by exploiting property (21), consider, the following two expressions relating the resulting velocity vectors of the three sub-chains end-effectors

$$\left( i \mathbf{v} - \mathbf{S}^3 i \mathbf{v} \right) = \left( i \mathbf{v} - \mathbf{v}_i \right) \mathbf{S} \left( \mathbf{S}^3 - \mathbf{v} \right) \quad ; \quad i = 1, 2 \tag{42}$$

and note that, since the contributions to the upper-lying end-effector velocities provided by the under-lying chain is always of rigid-motion type, in order to satisfy the kinematic constraints, we should have

$$\sum_{i=l}^2 \left( -1 \right) i \mathbf{v} \mathbf{S}^{-1} \left( i \mathbf{v} - \mathbf{v}_i \right) \mathbf{S} \left( \mathbf{S}^3 - \mathbf{v} \right) = 0 \tag{43}$$

(where for sake of convenience we have here expressed the kinematic constraints with respect to frame <e> intended as instantaneously belonging to the body frame rigid space) or, equivalently:

$$\sum_{i=l}^2 \left( -1 \right) i \mathbf{v} \mathbf{S}^{-1} \mathbf{v}_i \mathbf{S} \left( \mathbf{S}^3 - \mathbf{v} \right) \equiv U \left( \mathbf{S}^3 - \mathbf{v} \right) = 0 \tag{44}$$

3. At this point, since (44) clearly states the condition that must be satisfied by the vector $\left( \mathbf{S}^3 - \mathbf{v} \right)$ in order to guarantee kinematic consistency, we can consequently think to “force” the fulfillment of such condition at the beginning of the forward phase, by simply providing to the lower PCU of every upper-lying chain the following “safe” initial error vector

$$i \mathbf{v} = \mathbf{SP} \left( \mathbf{S}^3 - \mathbf{v} \right) ; \quad i = 1, 2 \tag{45}$$

where matrix

$$\mathbf{P} \equiv \left( I - U \mathbf{U} \right) \tag{46}$$

allows to consider just those component parts of vector $\left( \mathbf{S}^3 - \mathbf{v} \right)$ lying onto the kernel of matrix $U$, and therefore assuring the fulfillment of the rigid-body motion constraints.

As a consequence of such an imposition, it is easy to verify that relationships (42), result to be modified as follows

$$\left( i \mathbf{v} - \mathbf{S}^3 i \mathbf{v} \right) = \left( i \mathbf{v} - \mathbf{v}_i \right) \mathbf{S} \left( \mathbf{S}^3 - \mathbf{v} \right) \quad ; \quad i = 1, 2 \tag{47}$$

4. At this point, by substituting the expression of $i \mathbf{v}$ from (47) in the l.h.s. of (42), while keeping into account (21) we actually get (after same simple algebraic manipulations) the final form

$$\min_{i} \left( \sum_{i=l}^2 \left\| \mathbf{R} \left( \mathbf{v}_i \mathbf{S}^3 P + i \mathbf{S} \left( \mathbf{S}^3 - \mathbf{v} \right) \right) \right\|^2 \right) \tag{48}$$

Then, similarly to section 3, we could now finally think continuing minimizing (48) by now applying dynamic programming along the under-lying sub-chain; obviously upon transforming (48) itself into the associated standard equivalent form

$$\min_{i} \left\| 3 \mathbf{R} \left( \mathbf{S}^3 - \mathbf{v} \right) \right\|^2 \tag{49}$$
where $^3R$ matrix is obtained by the following factorization

$$^3Q = \sum_{i=1}^2 (V_i^iSP + iS(I-P))^TR_i^i \Rightarrow \begin{pmatrix}^3Q \end{pmatrix} = [^3R]^3R$$ (50)

A closed look to (48), once compared with its dual one for the unconstrained case, obtained by employing projected form (26) into expression (24); that is

$$\min_{\hat{q}} \left( \sum_{i=1}^2 \left[ R_i^iV_i^iS(\hat{x}_i^2 - \hat{x}_i^1) \right]^2 \right)$$ (51)

reveals very useful, since we can note how (48) can also be interpreted as (51) itself, once perturbed by a projection matrix $P \neq I$, expressing the existence of the grasping constraints.

As it can be now easily realized, relationships (40), (46) and finally factorization (50) all together represent the pre-processing activities to be in this case performed by the PCU located in correspondence of the Y-node, before allowing the underlying chain to run its own dynamic programming procedure. The required computational burden is clearly comparable with that for the tree-structured open chains, while also exhibiting a structural and procedural strictly close similarity.

**FURTHER WORKS**

Further work will primarily include a deeper investigation about the possibility of devising distributable optimal solutions even for the case of constrained tree-structured chains. Moreover, the extension of the developed theory to face also with the problem of joint limits and to allow contemporary execution of multiple tasks with priorities, is it also under current investigation. Finally an experimental prototype of modular arm implementing the proposed distributed kinematic inversion technique is currently at an advanced stage of development.

**REFERENCES**


